Vis Comput (2017) 33:801–810
DOI 10.1007/s00371-017-1389-2

ORIGINAL ARTICLE

Cloth compression using local cylindrical coordinates

Jiong Chen[1] · Yicun Zheng[1] · Ying Song[2] · Hanqiu Sun[3] · Hujun Bao[1] · Jin Huang[1]

Published online: 13 May 2017
© Springer-Verlag Berlin Heidelberg 2017

Abstract Dense triangular mesh is required to represent fine wrinkle details, which leads to heavy cost of storage and network transmission payload for cloth animation. This paper describes a simple and efficient compression method based on the nearly inextensible property of cloth, whose main degrees of freedom are the dihedral angles. Given a single frame as the reference, we build a local cylindrical coordinate system and encode the vertex as three channels: dihedral angle, change of radius and height w.r.t. the reference. The values of latter two channels are close to zero due to the inextensibility of cloth, which helps for a high compression ratio. Compared with previous approaches, our method can achieve a higher compression ratio with lower computational cost.

Keywords Data compression · Cloth animation · Strain limiting · Local cylindrical coordinate

1 Introduction

Virtual cloth has a wide range of applications and attracts a lot of attention in the computer graphics community. Although cloth simulation techniques and the computational ability of devices get largely improved in past decades, realistically reproducing the behavior of garments with fine details is still computational costly, which makes it impractical for mobile applications or those demanding for high frame rates. Thus, it is an attractive scheme for many applications (e.g., virtual reality, virtual try on and computer games) to compute the animation on server side and transfer the results to the end devices with limited computational ability for display only. In this circumstance, since a large number of vertices is required to represent delicate features such as wrinkles and torsional folds, it is important to encode the mesh sequence in a very efficient way: small data size, low decompression cost and low latency.

There is already extensive research on compression of mesh and animation. Most of them rely on the prior that the motion is smooth or nearly rigid for a local patch of nearby triangles. However, such a prior does not match the nature of virtual cloth and its animation well: strong stretch stiffness and weak bending stiffness make the triangles nearly rigid individually but have strong variance of rotation. Therefore the cloth animation generally exhibits complex and rapidly changed local details in a nonlinear manner and is challenging for those compression methods.

Our method is exactly motivated by the above facts that the triangles in the animated cloth are nearly rigid. The key idea is to build a local cylindrical coordinate system for each triangle of the animated mesh using a triplet: dihedral angle, changes of radius and height *respect to a reference mesh*. Because of the nearly inextensible property, only the dihedral angle may have significant variance, while the other two components are close to zero, which naturally provides a compact representation for cloth animation. To fully exploit the power of such a compact representation, we carefully analyze the relationship between the reconstruction error and the three channels of dihedral angle, radius and height increments, and

✉ Jin Huang
hj@cad.zju.edu.cn

Ying Song
ysong@zstu.edu.cn

1 Zhejiang University, Hangzhou, China

2 Zhejiang Sci-Tech University, Hangzhou, China

3 The Chinese University of Hong Kong, Shatin, N.T., Hong Kong

provide a method to automatically decide an optimal setting of number of bits for each channel.

Compared with previous techniques, our method achieves better compression ratio under comparable accuracy for common cases. The benefit becomes especially significant when the cloth gets involved in a violent responsive environment. Without complex process like multi-layer decompression [21,23], our approach is simple for implementation and faster in both compression and decompression.

2 Related work

Researchers have made much efforts on efficient and realistic cloth simulation. Besides complex collision, the nearly inextensible property also imposes big challenges for virtual clothing. For the consideration of stability, implicit integrator is usually adopted [4], but the numerical damping and the cost of solving a large-scale poor-conditioned linear system make it impractical for applications asking for high performance, realistic effects and complex collision handling. To improve the performance of the implicit integrator, Liu et al. [14] proposed projective approach to alternatively update the spring directions and vertex positions in a way very similar to [8]. Contrast to those penalty-based approaches, some methods use hard constraints on strain to remedy the problem of numerical instability [7,16,24]. Recently, Chebyshev semi-iteration[27] is used to accelerate the convergence of the above position-based and projective dynamics. However, this method currently cannot integrate with collision handling well, and runs at the risk of divergence. Some of the techniques [28,30], try to save the computational cost by synthesizing detailed wrinkles and folds, but the results are not always realistic. More discussion on virtual clothing can be found in a survey [25]. Roughly speaking, current approaches either lack enough efficiency or enough realism.

Compressing meshes or mesh sequences have also been explored before. People have proposed many successful techniques to compress the connectivity information [3,18]. Geometric data are often much larger than the connectivity one. The widely applied parallelogram prediction scheme [26] works well for nearly regular and flat regions. Better compression ratio could be achieved by using model-specific prior. For instance, Shikhare et al. [20] proposed to discover the similar patches in engineering models and remove such redundancy. Some other methods, like the spectral-based methods [11], apply a prior that the shape can be efficiently represented in a low-dimensional linear subspace spanned by the eigenvectors of mesh Laplacian. Similar priors were widely used in many compression methods, even for animation sequence. Alexa et al. [2] used Principle Component Analysis (PCA) on the shapes in the sequence. The method of [19] combines the PCA with the clustering methods to get localized bases. Methods based on skinning [10] also aim at constructing effective subspace (or, namely, pose space [13]). For better compression ratio, linear prediction coding has been used along with PCA [12], skinning [15], etc. However, cloth animations contain many *time-variant complex details*, which does not match the previous priors, and thus makes it very difficult to efficiently find similar patches or pre-compute an effective linear subspace. Our method also takes a prior, inextensibility of the mesh, which is one of the most significant property of clothes.

Some other animation compression methods mainly focus on designing effective predictors to explore the redundancy in temporal or spatial domain. By assuming the velocity field is spatially smooth, the technique of Dynapack [9] predicts the vertex offset by a linear combination of its neighboring ones. This method is improved in [21] by a spatially coarse to fine multi-layer predictor. Such methods assume that the animation is smooth in both spatial and temporal domain. Unluckily, such smoothness may be seriously violated in cloth animation due to complex collision, buckling and external forces. The complex shape and violent dynamics make it very difficult to design an efficient predictor. We also noticed that dihedral angle has also been used to encode animation [22], but this method assumes that the dihedral angles are almost constant from frame to frame, which contradicts to the obvious fact that the dihedral angles are the main DOFs for cloth animation. More recently, Stefanoski and Ostermann [23] proposed a scalable predictive coding (SPC) scheme, which decomposes the dynamic animation into multiple spatial and temporal layers, and employs prediction in the space of the rotation-invariant coordinates compensating the local rigid motion. Later, the compression ratio of SPC is improved by Bici and Akar [5] and Ahn et al. [1] with some novel predictive scheme or spatial decomposition methods. However, these methods do not fully explore the nature of cloth whose local rigidity assumption may not be desirably satisfied by the violent cloth, directly impacting the prediction accuracy of these methods. Besides, these methods all need a fairly complicated spatial decomposition algorithm, which involves intensive topological queries and modifications, and bring in extra difficulties for implementation. Also, if the animated mesh has a very high spatial resolution, the number of layers to be encoded is inevitably increasing, leading to efficiency loss on both storage and computation, especially not appealing for devices with limited capability like mobile phones.

3 Overview

A typical cloth simulation exhibits strong restrictions on the in-plane deformation. Therefore, we adopt the *Local Cylindrical Coordinate (LCC)* to represent the geometry, where

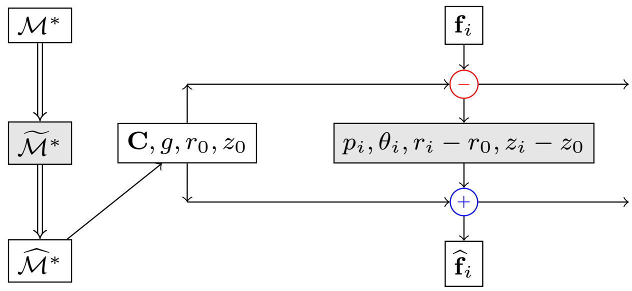


Fig. 1 The flowchart of our method. The components in *light gray* blocks are the final data of compression. The *arrows* in *double line* indicate the algorithm of Edgebreaker [18]

the dihedral angle component captures the main feature of an animated cloth, and the shear- and stretch-related components (radius and height) can be encoded by fewer bits.

Given a sequence of cloth animation composed of T meshes with the same connectivity $\mathbf{C}$, we denote its i-th frame by $\mathbf{f}_i = [\mathbf{v}_1^i, \mathbf{v}_2^i, \ldots, \mathbf{v}_N^i] \in \mathbb{R}^{3N}$, where N is the number of vertices and $\mathbf{v}$ stands for the three-dimensional vertex coordinate. The animation sequence is then represented by a $3N$-by-T matrix $\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_T] \in \mathbb{R}^{3N \times T}$. Each row of $\mathbf{F}$ corresponds to one-component trajectory of a vertex. The goal of our method is to efficiently compress such a matrix $\mathbf{F}$.

We first pick a reference mesh $\mathcal{M}^*$ with the same connectivity $\mathbf{C}$ and vertex coordinates $\mathbf{f}_0$, which could be the first frame of the animation, or a mesh isometric to the meshes in the animation, e.g., the rest configuration of the cloth simulation. This mesh is compressed into byte stream $\widetilde{\mathcal{M}}^*$ by Edgebreaker [18] or other algorithms for a single mesh, and decompressed back into a mesh $\widehat{\mathcal{M}}^*$.

As shown in Fig. 1, each frame in the animation is compressed and decompressed individually, which is easy for parallel processing and suitable for low-latency applications. Let us denote the spanning tree of mesh dual graph as g, while three components of LCC as θ, r, z, respectively. Then, each frame can be represented by the three vertex coordinates p of the root triangle in g, dihedral angles θ, the changes of radius r and height z in the local cylindrical coordinate system respect to the reference mesh $\widehat{\mathcal{M}}^*$. An optional step of temporal prediction can be used if the animation is smooth enough. After quantizing these data, an entropy encoding is applied to each frame for low-latency transmission and response. Of course, one can also apply entropy coding to all the data together for the highest compression ratio.

4 LCC-based coding of cloth meshes

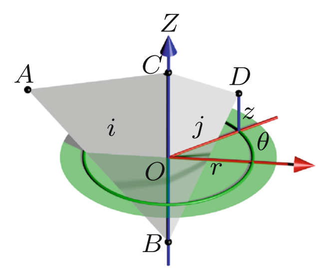
In this section, we will elaborate our algorithms in details.

4.1 Spanning tree

Our method shares similarity to many previous methods that each vertex is encoded based on the relative position to one of its adjacent triangles. In order to visit all vertices without repetition, we can build a spanning tree on the dual mesh and traverse the tree in a given order. Both the tree in Edgebreaker and a customized one built by users can be applied, and the selection has no significant difference on results according to the experiments. The tree and its traversal should be identical in both procedures of compression and decompression for all frames, which is easy to be satisfied by using a fixed tree construction and traversal strategy based on the connectivity only.

Starting from the root triangle, we first directly encode its three vertex positions p, because geometry of the mesh can only be determined by the dihedral angle, radius and height up to a globally rigid transformation. Then, each vertex except for the three of the root triangle will be encoded just once during the traversal, which leads to $N - 3$ triplets (θ, r, z), i.e., the LCC for the vertex.

4.2 Local cylindrical coordinates



Without loss of generality, suppose we are visiting face j from face i in the same frame bounded by vertices A, B, C (see the inset). A local cylindrical coordinate system can be built according to the triangle $\triangle ABC$: the origin O is set to be the middle of the pivoting edge $\overline{BC}$, whose counter-clockwise direction in face i is the longitudinal axis direction Z (blue arrow), and the polar axis (red arrow) is taken as the cross product of Z and the normal of face i.

Under this coordinate system, it is a trivial task to convert the Cartesian coordinates of vertex D into the LCC (θ, r, z). To make radius and height components invariant under uniform scaling, we divide these two components by the average edge length L of decompressed reference mesh $\widehat{\mathcal{M}}^*$. Such a normalization also makes the unit change of angle (θ) value and the length (r, z) value results in comparable magnitude of shape change. This nonlinear coordinates transformation actually helps to orthogonally encode the in-plane and out-of-plane local deformation. Most cloth animation follows the inextensible property, so the radius and height components have little variance. Figure 2 shows the distribution

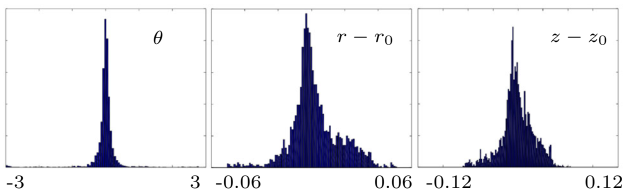


Fig. 2 The change of radius and height components fall in a much smaller range compared with the angle component

of cloth animation (windy flag) with strain limited in range [0.95, 1.05].

In other words, the information in terms of shape error contained in radius and height components is much smaller than the angle component, which implies that under the same precision requirement, we can use fewer bits to quantize those components.

4.3 Quantization

In compression, we use a simple linear quantization function $q(x) = \text{floor}(2^b x/\sigma)$ to quantize the floating number, where σ is the absolute maximal approximate range of x and b is the bit number used for quantization. The range of dihedral angle can be directly set as $[-\pi, \pi]$, and to get the ranges of other two channels of LCC, we can simply run the coding algorithm for a few frames ahead to get the approximate ranges.

To automatically find a good setting of bits for the channels, we need to estimate the error related to unit loss in each channel. Under the assumption that the mesh is nearly uniform with average height L, which is usually true for high-resolution detailed animation, the error per radian is just L. Because we have divided r and z by L, the error of unit loss in radius and height loss is also L. Therefore, to balance the error per loss for the three channels, if the number of bits for the dihedral angle channel is b_θ, we set the number of bits b_{rh} for both radius and height channels as

$$b_{\text{rh}} = b_\theta - \lfloor 0.5 - \log_2(\sigma) \rfloor, \tag{1}$$

where σ is the max of approximate range of radius and height increments. From Fig. 3, one can *roughly* verify the above equation. Beyond the above-recommended bits, the error will not decrease significantly and vice versa.

This equation also indicates that the compression efficiency of our algorithm is tightly related to the strain limiting. Suppose that the cloth is very close to developable, e.g., taking the result from [6] as the input, we can use only dihedral angles along with the root face positions to accurately encode the geometry. Under loose strain limiting, the values of radius and height components will distribute in a relatively larger range, thus more bits are required for them.

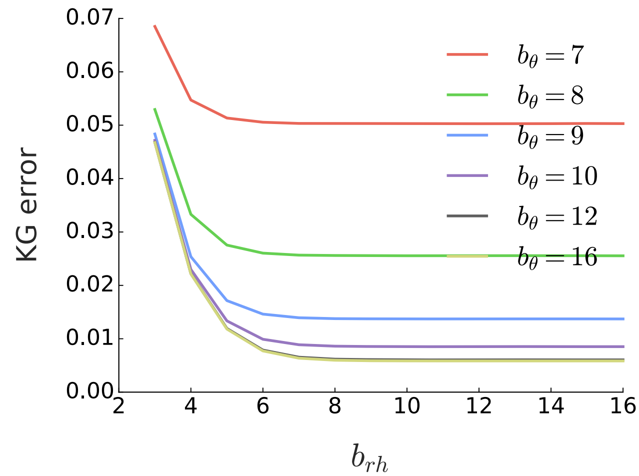


Fig. 3 Fixing b_θ, we change the b_{rh} and calculate the reconstruction error (defined in Eq. 2) of windy flag sequence with strain limits [0.95, 1.05]. In this example, σ is found to be 0.0886128 thus the recommended bits difference is 3

Straightforward implementation of the above method will lead to significant visual artifacts at the crossings of two traversal paths because of the accumulated quantization error. To address the problem, we propose to decompress the mesh simultaneously along with the compression procedure. To be more precise, when reaching an unvisited vertex during compression, we evaluate and quantize its LCC, then immediately compute the reconstructed vertex position. This reconstructed vertex position will be used in the all of the following local cylindrical coordinate system constructions. This way guarantees that the decompression procedure always uses an identical local coordinate system as compression. From another point of the view, the error of the local cylindrical coordinate system caused by quantization will be compensated by the difference of LCCs computed under the different coordinate systems. The above scheme successfully avoids accumulated error even the depth of the tree is large.

4.4 Temporal prediction

When the animation sequence is smooth enough, our method can be also easily combined with temporal prediction method to further improve the compression rate. For simplicity, we just apply the temporal prediction method proposed in [23] to the coordinates generated by our LCC spatial coding scheme. Their method builds a temporal prediction hierarchy and uses the neighboring frames to predict the middle one recursively, where each encoded LCC channel x_i in frame m are given by $\Delta x_i^{(m)} = x_i - (\hat{x}_i^{(l)} + \hat{x}_i^{(r)})/2$, where $\hat{x}_i^{(l)}$ and $\hat{x}_i^{(r)}$ are the reconstructed values of LCC components of previous and post-neighboring frames. However, when the animation becomes violent, applying such temporal prediction scheme could instead enlarge the variance of angle component with doubled range of $[-2\pi, 2\pi]$. In such circumstance, the compression ratio may be lowered down compared to the case without temporal prediction (see Fig. 8). As shown in the accompanying video, for physically plausible 25 fps animation, the motion of cloth is usually not very smooth. Apart from losing such benefits for compression, the temporal prediction will also limit the flexibility and efficiency for random access of the sequence, which is proven to be important in some applications like fast forward and random access of the animation sequence. Therefore, the temporal prediction is optional.

5 Implementation details and results

In this section, we will first introduce the details of our implementation, and then show the results and comparisons.

5.1 Implementation details

To obtain the animation sequences, we use ARCSim [17, 29] without adaptive remeshing for cloth simulation. The simulator performs a post-processing after the step of implicit time integration to enforce strain limiting, which is controlled by a prescribed parameter.

The pre-transmitted mesh $\widehat{\mathcal{M}}^*$ taken as the reference can be either the rest configuration for simulation or the first frame of the sequence. Since a flat plane can be very efficiently compressed by Edgebreaker algorithm, to save the storage for the reference mesh, we choose a planar one if it is available and isometric to meshes in sequence. Otherwise, for simplicity, we pick the first frame as the reference one, and encoding the remaining frames using our method. Indeed, for a relatively long sequence, the choice of the reference mesh is not that important.

Since our coding algorithms are associated with a simple tree traversal on the dual mesh, generating the spanning tree is the first step before the actual encoding and decoding algorithms. For simplicity, we assign uniform weights on all edges of the dual graph, extract the minimal spanning tree. In such configuration, the experiments indicate that different traversal orders (depth-first or width-first) have no significant impact because the difference of compression ratio is usually less than 5%. In our implementation, depth-first traversal is adopted. For those highly irregular meshes, it is possible to incorporate the geometry information of the reference mesh to build a better structured tree and explore an optimal traversal order for more efficient compression.

Our framework can be performed in both download and stream mode for higher compression ratio and lower latency, respectively. For the former mode, the encoder collects the byte streams from all frames and compresses the whole. For the later one, it just compresses the data frame by frame. Although the download mode is able to achieve higher compression ratio (about 15%), the decompression procedure asks more memory than the stream mode, which may be not desired in some applications running on a device with limited resources.

5.2 Error metric

In order to measure the difference between the original sequence and the reconstructed one, we adopt the well-known *KG error* proposed in [12] as the quality metric for compression. The error is defined as,

$$e_{KG} = 100\frac{\|\mathbf{F} - \widehat{\mathbf{F}}\|}{\|\mathbf{F} - \mathbf{E}(\mathbf{F})\|}, \tag{2}$$

where $\|.\|$ is the Frobenius norm, $\mathbf{F}$ is a $3N \times T$ matrix encoding the geometry of the input mesh sequence and $\widehat{\mathbf{F}}$ is the reconstructed sequence with the same size as $\mathbf{F}$. $\mathbf{E}(\mathbf{F})$ is an average matrix whose i-th column is

$$[\bar{x}_i, \bar{y}_i, \bar{z}z_i, \bar{x}_i, \bar{y}_i, \bar{z}z_i, \ldots, \bar{x}_i, \bar{y}_i, \bar{z}z_i]^T \in \mathbb{R}^{3n},$$

where $\bar{x}_i$ is the mean value of all x coordinates in frame i, and the same convention holds for $\bar{y}_i$ and $\bar{z}_i$. To demonstrate the relationship between KG error and visual quality, we sample several frames from two reconstructed mesh sequences with error magnitude ranging from 0.05 to 1. From Fig. 4, one can see that results with error below 0.1 are almost visually indistinguishable from the input one.

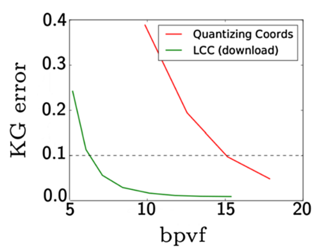
5.3 Comparisons

We compare our LCC-based coding with several widely used methods for compressing a single mesh and animation sequences. First of all, we compare our method with the most intuitive way to compress the sequences by just simply quantizing the Cartesian coordinates the vertices. Next, we apply subspace method to encode the sequences and illustrate its poor efficiency when applied on cloth animation. Finally, we compare our method with several predictive coding methods, including single layer methods Edgebreaker [18], Dynapack [9] and hierarchical method SPC [23]. Notice that our algorithm can be performed in either streamable or download mode.

We use two sequences as the benchmarks (see Fig. 5 top two rows) in our experiment. One is an animation sequence of a windy flag (4225 vertices, 8192 faces) with time-variant wind forces exerted on the triangle faces. The other consists of a piece of cloth (16641 vertices, 32768 faces) falling onto a solid ball, involving frictions and collisions. Same to [23], we also take *bits per vertex per frame* (bpvf) to characterize the compression ratio. For the uncompressed animation

Fig. 4 The visual quality under different KG errors

data, each vertex has three Euclidean coordinates recorded by single-precision floating numbers, thus the bpvf of the raw animation sequence is 96.



As shown in the inset, comparing to directly quantizing the Cartesian coordinates, our result is about three times better on windy flag benchmarks, which matches the previous analysis about the degrees of freedom in nearly inextensible clothes.

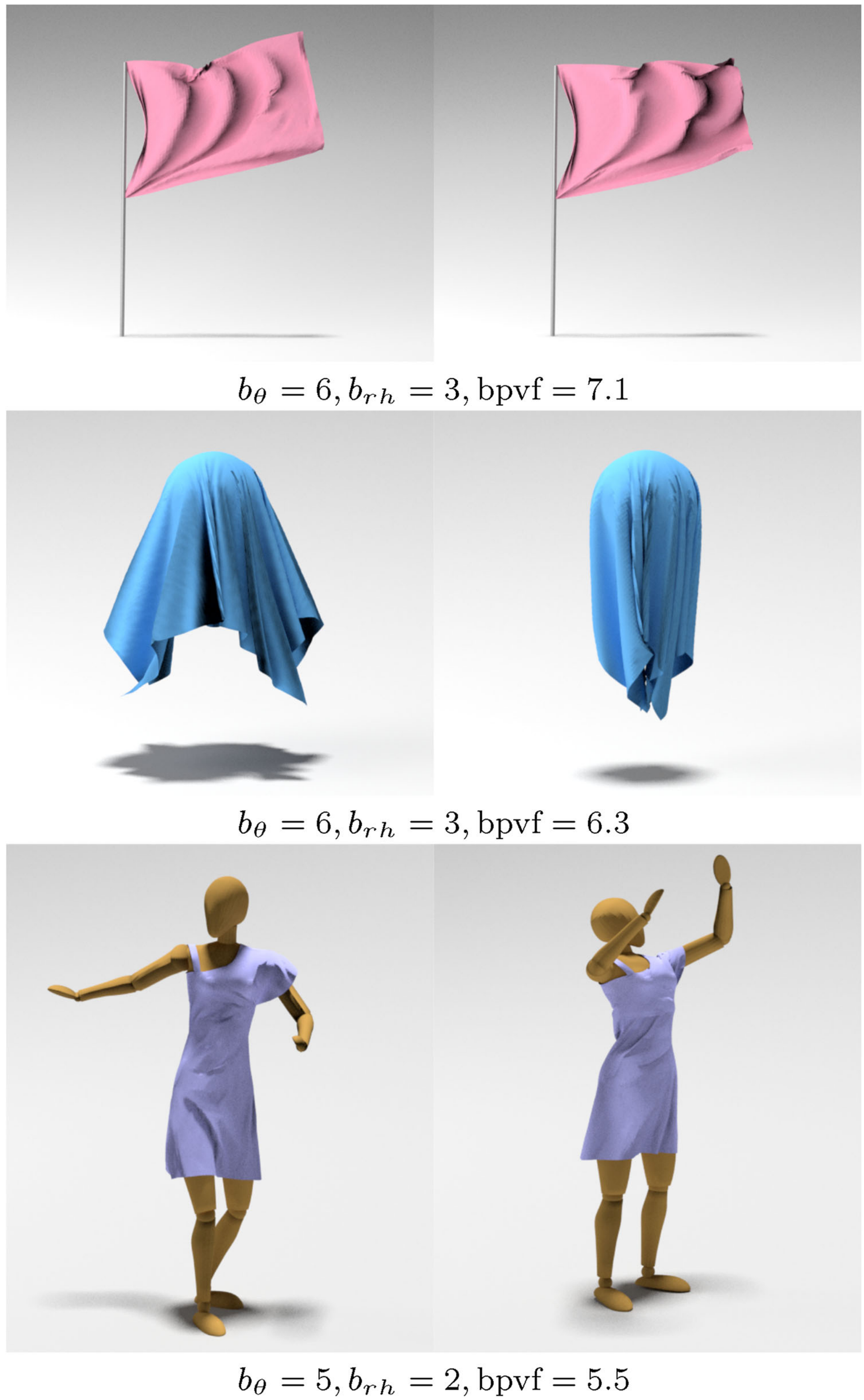
We also apply the PCA-based method [2] to construct a linear subspace for the sequence of a windy flag. Without applying any quantization to the basis vectors $\mathbf{U}$ and the corresponding coefficients $\mathbf{q}$, we calculate the error of the reconstructed result $\widehat{\mathbf{F}} = \mathbf{U}\mathbf{q}$ and plot the error curves w.r.t. the varying number of bases. As shown in Fig. 6, the reconstruction error does not decrease significantly as increases the number of bases. Even using two-third of the full space dimension (300 in this example), the KG error is still larger than 0.2. If quantization is applied, it will be worse obviously. Although we do not make comparisons with other subspace methods, e.g., skinning-based methods [10,15], one can expect that finding a small set of prescribed linear bases to capture complex shapes in cloth animation is extremely challenging.

We also compared our method with more complicated modern techniques. As shown in Fig. 7, even under stream mode and without temporal prediction, our method outperforms both Edgebreaker and Dynapack significantly. When $e_{KG} \approx 0.1$, which is a typical tolerance of precision loss, our compression ratio is two times higher than theirs. It is interesting to see that Dynapack has similar performance to Edgebreaker in windy flag example where the motion is fairly violent, although it intensively uses the temporal coherence. It is very likely that the cloth animations exhibit complex and violent dynamics due to external forces and collisions, which makes the linear predictor fail. Contrary to Dynapack, our method without temporal prediction does not directly use the temporal coherence, but the LCC naturally and automatically discovers the invariance of radius and height in cloth animations.

We created more scenarios to compare our method with SPC. In order to fully explore the performance of our algorithm under different temporal smoothness of animation, we generated a long animation sequence with small timestep of windy flag consisting of 8192 frames via ARCSim simulator. To obtain the sequences with different temporal resolution, we uniformly sample the animation with different stride. For simplicity, the temporal resolution of the resulting subsequence is denoted by logarithm of downsampled frames number, e.g., the sequence with 8192 frames has temporal resolution 13. The higher the temporal resolution is, the smoother the animation sequence will be. Under different temporal resolution, we compared the performance of SPC method and our method with and without temporal prediction, and the results are shown is Fig. 8.

When the animation is relatively violent (small temporal resolution), our method without temporal prediction (labeled as LCC) still achieves high compression ratio, while SPC method has significant efficiency loss in such circumstance. Applying temporal prediction in our method (labeled as LCC.T) will improve the compression ratio when the temporal resolution is larger than 8. Under KG error of 0.1, our method (LCC or LCC.T) obviously surpasses SPC until temporal resolution goes to 12, which is a very slow and smooth motion. In our experiments, sequence with temporal resolu-

Fig. 5 Reconstructed results with KG error about 0.1



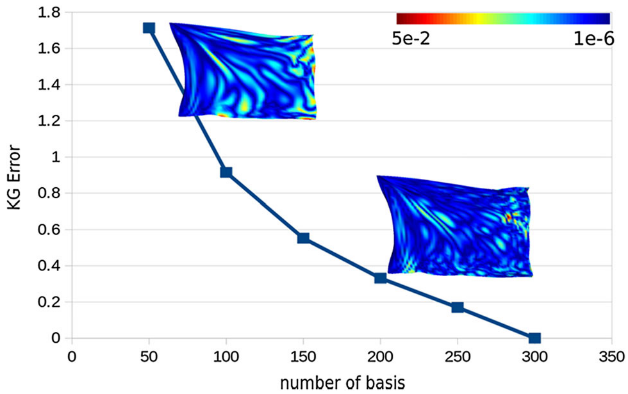


Fig. 6 Complex wrinkles cannot be efficiently represented by smooth PCA bases. The error distribution is shown for results using 50 and 200 bases

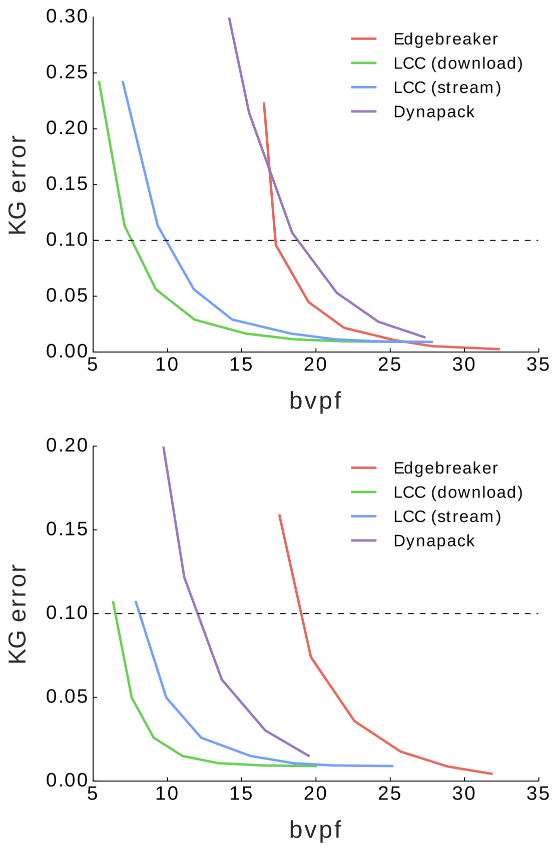


Fig. 7 The bpvf-KG curve of several methods on windy flag benchmark (*top*) and cloth falling benchmark (*bottom*)

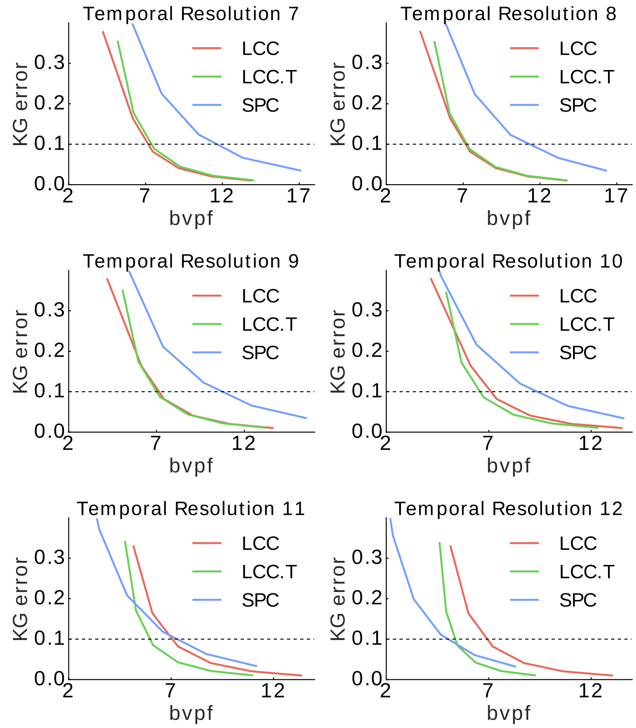


Fig. 8 bpvf-KG curve of our method (LCC, LCC.T) and SPC under different temporal resolution

tion between 9 and 10 will generate the most plausible results when played as video in 25 fps (see the accompanying video). Therefore for most cloth animation with moderate and violent motion, our method is more attractive.

5.4 Timing

When using our method without temporal prediction, the data for each frame are independent, thus the frames in an animation can be easily compressed and decompressed in parallel (except the step of entropy coding under the download mode). Like many other predictive methods, the major operation of our algorithm is to traverse a spanning tree and apply some coordinate transformation. Thus to offer a fair comparison, we measure the timing of this step without applying any parallel acceleration.

As shown in Table 1, the compression and decompression algorithms exhibit similar performance and can compress (decompress) about 1k vertices in a millisecond. This is simply because both algorithms are only slightly different in the coordinate transformation step during the tree traversal. Obviously, the complexity of the algorithm is linear to the number of vertices. For a highly detailed animation (dress) using over 100 k vertices per frame, less than 100 ms is enough for each frame.

According to the statistics in the paper [23], SPC achieves 46 fps decoding a model with about 16k vertices, while our method can get about 100 fps under comparable mesh size, which is two times faster than SPC in this scale.

6 Conclusion and future work

This paper has proposed a new representation for cloth animation based on local cylindrical coordinates (LCC). LCC is suitable for cloth animation compression because

Table 1 Timing for geometry compression (comp.) and decompression (decomp.) on a workstation equipped with a 12-core Intel(R) Xeon(R) E5-2609 CPU with 1.90 GHz

Animation	Windy flag	Cloth falling	Dress
#vert	4225	16641	104482
#frames	300	250	300
comp. tims (s)	1.13	3.61	32.0
decomp. time (s)	1.0	2.6	27.0
comp. #vert/ms	1118	1154	980
decomp. #vert/ms	1268	1600	1161

Just one thread is used

the dihedral component of LCC describes the main feature of the animation sequence, while the radius and height components are nearly constant, which implies very strong temporal coherence that a few bits are enough for their quantization. Straightforwardly applying LCC would cause significant error accumulation. To solve this problem, we compensate the error when calculating the coordinates using a simultaneously decompressed mesh. Based on the analysis or error per loss in quantization, we also provide a scheme to automatically decide the setting of bits for different channels. Experimental results show that the proposed method outperforms many methods even without delicately exploiting the temporal coherence because our method automatically discovers the strong temporal coherence of radius and height. When the cloth animation is fairly smooth, the temporal prediction can help to further improve the compression rate as our experiments indicate.

One limitation is that our method cannot exploit the spatial coherence, which means that the compression ratio may be low if the triangles are much smaller than the granularity of wrinkles. Besides, our method currently cannot automatically determine the optimal bits for encoding the LCC channels when a certain KG error is given as the upper bound. Running much more test cases and analyze the bits-error curves may help to obtain some practical guidance. Lastly, we just use a simple linear temporal predictor to exploit the temporal coherence. Although in many applications, the spatial and temporal resolutions of mesh sequence will be minimized under acceptable visual quality for the sake of efficiency, it is still an interesting future work to find effective spatial and temporal predictors based on the characteristics of dihedral angle component for general cloth animation.

Acknowledgements We would like to thank the anonymous reviewers for their valuable comments and suggestions. This work is partially supported by NSFC (Nos. 61522209, 61210007). Prof. Ying Song is supported by NSFC (No. 61602416) and Prof. Hanqiu Sun is supported by RGC research grant (Ref. 416212), UGC grant (No. 4055060), NSFC funds (Nos. 61379087, 61602183).

References

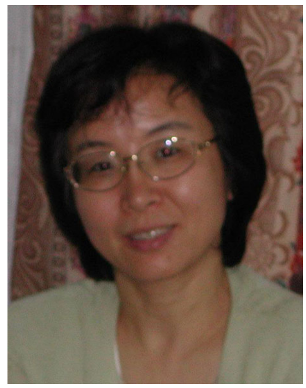
1. Ahn, J.K., Koh, Y.J., Kim, C.S.: Efficient fine-granular scalable coding of 3D mesh sequences. IEEE Trans. Multimedia **15**(3), 485–497 (2013)
2. Alexa, M., Mller, W.: Representing animations by principal components. Comput. Graph. Forum **19**(3), 411–418 (2000)
3. Alliez, P., Desbrun, M.: Valence-driven connectivity encoding for 3D meshes. Comput. Graph. Forum **20**(3), 480–489 (2001)
4. Baraff, D., Witkin, A.: Large steps in cloth simulation. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, pp. 43–54. ACM (1998)
5. Bici, M.O., Akar, G.B.: Improved prediction methods for scalable predictive animated mesh compression. J. Vis. Commun. Image Represent. **22**(7), 577–589 (2011)
6. English, E., Bridson, R.: Animating developable surfaces using nonconforming elements. ACM Trans. Graph. (TOG) **27**(3), 1–5 (2008)
7. Goldenthal, R., Harmon, D., Fattal, R., Bercovier, M., Grinspun, E.: Efficient simulation of inextensible cloth. ACM Trans. Graph. (TOG) **26**(3), 49 (2007)
8. Huang, J., Zhang, H., Shi, X., Liu, X., Bao, H.: Interactive mesh deformation with pseudo material effects. Comput. Animat. Virtual Worlds **17**(3–4), 383–392 (2006)
9. Ibarria, L., Rossignac, J.: Dynapack: space-time compression of the 3d animations of triangle meshes with fixed connectivity. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 126–135. Eurographics Association (2003)
10. James, D.L., Twigg, C.D.: Skinning mesh animations. ACM Trans. Graph. (TOG) **24**(3), 399–407 (2005)
11. Karni, Z., Gotsman, C.: Spectral compression of mesh geometry. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '00, pp. 279–286. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)
12. Karni, Z., Gotsman, C.: Compression of soft-body animation sequences. Comput. Graph. **28**(1), 25–34 (2004)
13. Lewis, J.P., Cordner, M., Fong, N.: Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '00, pp. 165–172. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)
14. Liu, T., Bargteil, A.W., O'Brien, J.F., Kavan, L.: Fast simulation of mass-spring systems. ACM Trans. Graph. (TOG) **32**(6), 214:1–214:7 (2013)
15. Mamou, K., Zaharia, T., Prêteux, F., Stefanoski, N., Ostermann, J.: Frame-based compression of animated meshes in MPEG-4. In: 2008 IEEE International Conference on Multimedia and Expo, pp. 1121–1124. IEEE (2008)
16. Müller, M., Heidelberger, B., Hennix, M., Ratcliff, J.: Position based dynamics. J. Vis. Commun. Image Represent. **18**(2), 109–118 (2007)
17. Narain, R., Samii, A., O'Brien, J.F.: Adaptive anisotropic remeshing for cloth simulation. ACM Trans. Graph. (TOG) **31**(6), 152 (2012)
18. Rossignac, J.: Edgebreaker: connectivity compression for triangle meshes. IEEE Trans. Vis. Comput. Graph. **5**(1), 47–61 (1999)
19. Sattler, M., Sarlette, R., Klein, R.: Simple and efficient compression of animation sequences. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 209–217. ACM (2005)
20. Shikhare, D., Bhakar, S., Mudur, S.P.: Compression of large 3d engineering models using automatic discovery of repeating geo-

metric features. In: Proceedings of the Vision Modeling and Visualization Conference 2001 (VMV-01), Stuttgart, Germany, November 21–23, 2001, pp. 233–240 (2001)
21. Stefanoski, N., Liu, X., Klie, P., Ostermann, J.: Scalable linear predictive coding of time-consistent 3D mesh sequences. In: 3DTV Conference, 2007, pp. 1–4. IEEE (2007)
22. Stefanoski, N., Ostermann, J.: Connectivity-guided predictive compression of dynamic 3d meshes. In: 2006 IEEE International Conference on Image Processing, pp. 2973–2976. IEEE (2006)
23. Stefanoski, N., Ostermann, J.: SPC: fast and efficient scalable predictive coding of animated meshes. In: Computer Graphics Forum, vol. 29, pp. 101–116. Wiley Online Library (2010)
24. Thomaszewski, B., Pabst, S., Straer, W.: Continuum-based strain limiting. Comput. Graph. Forum **28**(2), 569–576 (2009)
25. Thomaszewski, B., Wacker, M., Straβer, W., Lyard, E., Luible, C., Volino, P., Kasap, M., Muggeo, V., Magnenat-Thalmann, N.: Advanced topics in virtual garment simulation. In: Myszkowski, K., Havran, V. (eds.) Eurographics 2007—Tutorials. The Eurographics Association (2007)
26. Touma, C., Gotsman, C.: Triangle mesh compression. In: Proceedings of the Graphics Interface 1998 Conference, June 18–20, 1998, Vancouver, BC, Canada, pp. 26–34 (1998)
27. Wang, H.: A chebyshev semi-iterative approach for accelerating projective and position-based dynamics. ACM Trans. Graph. (TOG) **34**(6), 246:1–246:9 (2015)
28. Wang, H., Hecht, F., Ramamoorthi, R., O'Brien, J.F.: Example-based wrinkle synthesis for clothing animation. ACM Trans. Graph. (TOG) **29**(4), 107:1–107:8 (2010)
29. Wang, H., O'Brien, J.F., Ramamoorthi, R.: Data-driven elastic models for cloth: modeling and measurement. ACM Trans. Graph. (TOG) **30**(4), 71 (2011)
30. Xu, W., Umentani, N., Chao, Q., Mao, J., Jin, X., Tong, X.: Sensitivity-optimized rigging for example-based real-time clothing synthesis. ACM Trans. Graph. (TOG) **33**(4), 107:1–107:11 (2014)

Jiong Chen is currently a Ph.D. candidate in the State Key Laborate of CAD&CG, Zhejiang University, Hangzhou, China. His research interest is physically based simulation.

Yicun Zheng is an undergraduate student in School of Computer Science and Technology, Zhejiang University, HangZhou, China.

Ying Song is a full-time Lecturer in Computer Science at Zhejiang Sci-Tech University, China. She received B.S. degree and Ph.D. degree in Computer Science, both from Zhejiang University. Her research is concerned with various aspects of material appearance modelling (texture and reflectance), GPU hardware-accelerated rendering and related software development.



Hanqiu Sun received B.S. in electrical engineering from Huazhong University of Science and Technology, M.S. in electrical engineering from University of British Columbia, and Ph.D. in computer science from University of Alberta, Canada. She is currently an associate professor in CS&E Department of CUHK. Her current research interests include virtual reality, interactive graphics/animation, real-time and hybrid VR environments, computer-assisted surgery, mobile image/video synopsis and navigation, Haptics and dynamics simulations.

Hujun Bao received the B.S. and Ph.D. degrees in applied mathematics from Zhejiang University, Hangzhou, China, in 1987 and 1993, respectively. He is a Cheung Kong professor in the School of Computer Science and Technology in Zhejiang University. His research interests include geometry computing, vision computing, real-time rendering, and virtual reality.

Jin Huang received the Ph.D. degree from Computer Science Department, Zhejiang University, Hangzhou, China, in 2007, with Excellent Doc- toral Dissertation Award of China Computer Federation. He is a professor in the State Key Lab of CAD&CG of Zhejiang University. His research interests include geometry processing and physically based simulation.